Enhancing Social Interaction with Seamless Face Recognition on Google Glass: *Leveraging opportunistic multi-tasking on smart phones*

Shue-Ching Chia
Institute for Infocomm Research
1 Fusionopolis Way, Singapore 138632
scchia@i2r.a-star.edu.sg

Liyuan Li
Institute for Infocomm Research
1 Fusionopolis Way, Singapore 138632
lyli@i2r.a-star.edu.sg

Bappaditya Mandal
Institute for Infocomm Research
1 Fusionopolis Way, Singapore 138632
bmandal@i2r.a-star.edu.sg

Joo-Hwee Lim
Institute for Infocomm Research
1 Fusionopolis Way, Singapore 138632
joohwee@i2r.a-star.edu.sg

Qianli Xu
qxu@i2r.a-star.edu.sg

MobileHCI '15 Adjunct, August 24-27, 2015, Copenhagen, Denmark
ACM 978-1-4503-3653-6/15/08.
http://dx.doi.org/10.1145/2786567.2793697

Abstract
Wearable devices offer immense opportunities in both consumer and enterprise domains due to the hands-free interaction modality and the ability to provide information in real-time. However, due to hardware limitations, it presents a notable challenge to complex applications that have stringent demands on computational efficiency. Leveraging on the computing power of a connected mobile device, we propose a new multi-threaded asynchronous structure to implement opportunistic multi-tasking. We demonstrate an application of the structure for seamless face recognition for social interactions. The experimental studies show that the proposed structure can achieve better performance than a sequential synchronous one. The proposed method can be extended to similar applications in wearable devices.

Author Keywords
Wearable devices; face recognition; multi-threaded; asynchronous.

ACM Classification Keywords
C.5.3 Computer System Implementation: Microcomputers; D.1.3 Programming Techniques: Parallel Programming.

Introduction

In recent years, there has been a radical shift from traditional desktop computing to mobile computing and now to wearable computing [7]. By providing real-time relevant information and enabling hands-free operations, wearable computing provides immerse opportunities for various applications, such as face recognition for social interaction assistance. However, since wearable devices have limited hardware capabilities [1], it is challenging to run complex calculations such as image processing efficiently on these devices. Meanwhile, the very spirit of wearable computing is to maximize information accessibility, often requiring real-time response. As such, developers have been constantly challenged by the difficulty of providing optimal user experience with limited computational resources. To tackle this issue, many applications on wearable devices, e.g. wearable vision, employ a client-server structure, where the wearable device acts as an input/output (I/O) client, and the server sustains intensive computations [1, 3, 7]. The server is typically in the form of a wired tablet/smart phone or an infrastructure in the cloud. Nevertheless, such a structure put great demand on data flow and task synchronization between the two ends, failing which will cause considerable loss of efficiency. On traditional server and desktop workstations, multi-threading has been widely adopted to improve the performance of applications and user experience [2]. Borrowing such an idea, we propose a new multi-threaded asynchronous structure that effectively balances the computational load for data transfer and image processing on a platform consisting of a wearable camera and a mobile phone. We show that the proposed structure outperforms the traditional single-threaded one by leveraging the opportunistic multitasking strategy, and hence it leads to better user experience.

Sequential Synchronous Structure

A face recognition application was proposed in [5] using wearable devices for assisting social interactions. In this application, a client-server structure is adopted whereby intensive computations are offloaded to the server while the client acts as a dummy terminal for acquiring images and displaying results. This structure is implemented using Google Glass as the client and mobile phone as the server.

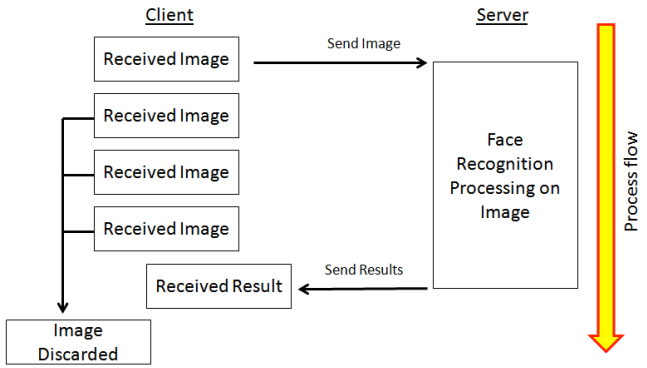

Figure 1: Process flow of the current structure. The client receives images from the camera live feed and sends them to the server. The server sends back the result after processing. During the recognition process on the server, all intermediate images from the camera are lost.

The client and server are connected via a single Bluetooth communication channel and all data exchange is conducted via this Bluetooth channel. The entire process is executed in a sequential synchronous manner as shown in Figure 1.

1. The client acquires images through its camera live feed and sends the images to the server for face recognition.
2. After performing the recognition, the server will send the results back to the client.
3. During the recognition process, all intermediate images received on the client are discarded.

The efficiency of the client-server structure relies heavily on task synchronization and data transmission between the two devices. As such, the current structure is less efficient because within a face recognition cycle, only one image is used for the recognition process while subsequent intermediate images are ignored. In a natural social interaction, there exist many factors that causes image-degradation, such as poor lighting conditions, presence of motion blur due to physical movements of the glass wearer and the target person. In addition, the camera on Google Glass is a wide-angle camera [4], resulting in small image footages of the faces. In real applications, the wearer may encounter undesirable delays, because the application is unable to obtain good recognition performance due to the poor image quality.

Multi-threaded Asynchronous Structure

To achieve high system performance and seamless user experience, we propose a new multi-threaded asynchronous structure to replace the existing sequential synchronous one. The key insight is *to exploit the power of parallel processing on multiple threads, so as to retain and process the intermediate images instead of discarding them*. By including these images for processing, we seek to increase the probability of getting successful recognition and improving the response time of the application.

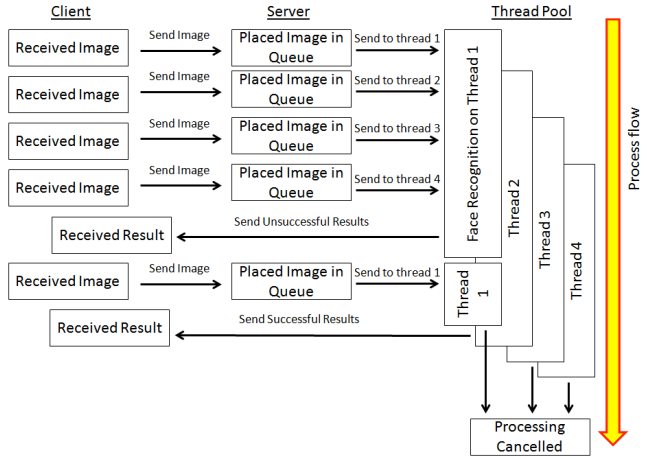

Figure 2: Process flow for the improved structure. Images are continuously sent to server, and are placed in a queue to be processed by threads. Recognition results are sent back to the client in an asynchronous manner. Upon a successful recognition, all pending images in the queue will be cancelled.

Figure 2 shows the process flow using the new structure. The points below are the key improvements made in the new structure.

1. Communication between the client and server is changed from synchronous to asynchronous. That is, the client will continuously send images from the camera live feed to the server without having to wait for the recognition results.

Figure 3: The target face from a data set with good lighting condition (top) and poor lighting condition (bottom).

2. In order to process the influx of incoming images, the processing on the server side is changed from single threaded to multi-threaded. We implement a thread pool, a queue and a scheduler on the server for multi-threaded processing. The number of threads created is equivalent to the number of CPU cores in the server and each thread performs recognition on the images. New images received by the server will be added to the queue and the scheduler is responsible for invoking idle threads to process pending images in the queue. After getting the recognition result, each thread will send the result back to the client.
3. As the structure is now asynchronous, the client must be able to send images and receive results at the same time. As this is impossible on a single Bluetooth channel, two separate Bluetooth channels are created instead. The client sends images to the server through channel 1; and the server sends back recognition results through channel 2.
4. Upon a successful recognition, all pending images in the queue are cancelled. To facilitate correct cancellation, a unique identifier, i.e. current timestamp in milliseconds, is used to group all images under the same recognition attempt. The client will generate this identifier and transmit it to the server, together with the image. Based on this identifier, the server is able to correctly cancel all related images in the queue.

Experiment I – Performance Test

To evaluate the performance of the new multi-threaded asynchronous processing structure (denoted as *M-Asyn*), we compare it with the original sequential synchronous processing structure (denoted as *S-Syn*). Both *M-Asyn* and *S-Syn* are implemented on Google Glass XE22 as the client and Samsung Galaxy Note 4, on Android 4.4.4 as the server. Leveraging on the quad-core processor, the system runs four threads in the *M-Asyn* mode.

Test Data

First, we trained our face recognition model on a dataset consisting of image sequences of 15 volunteers' faces. Note that the face recognition algorithm is independent of the communication mechanism between the wearable client and mobile server. Therefore, the performance difference between *M-Asyn* and *S-Syn* modes is caused by the structure itself rather than the face recognition algorithm.

To enable fair evaluation of both structures, we prepared 60 videos captured from 4 persons among the 15 volunteers. The video footages show human faces in a social interaction setting, with some degree of head motions. Both *M-Asyn* and *S-Syn* used the same input data to measure the performance of face recognition. The videos ranged from 4 to 5 seconds and were recorded at 30 frames per second. The video data were discretized into gray scale images. We manually inspected the videos and assign them into two categories based on the lighting condition. Half of the videos (30) have good lighting condition, while the rest (30) have poor lighting condition. The poor lighting condition is quite common in real social interaction scenarios when the target faces are subject to silhouetted or insufficient illumination. Figure 3 shows a face in good and poor lighting conditions.

Experiment and Performance Matrices

We modified the code of the image-sending module so that it transmits the pre-recorded images to simulate the camera live feed. To ensure precise time measurement, at each image transmission time point,

we used the time elapsed, in milliseconds, to derive the corresponding pre-recorded image number. For each data set, we executed *M-Asyn* and *S-Syn* 10 times, respectively, and recorded the following.

1. Percentage of successful recognitions among the 10 executions. A successful recognition means that the target person is recognized during the streaming of the test video.
2. Average timing taken for a successful recognition. This is measured as the duration from the start of each execution to the return of recognition result to the client. Only the timings for successful recognitions are averaged out.
3. Number of recognition attempts needed before a successful recognition. This is equivalent to the number of images processed by the server. For the *M-Asyn* mode, it is the sum of attempts made by multiple CPUs.

Results

The statistical results of the three performance measures are shown in Figure 4. We consider two independent variables: (1) the mode of structure (*M-Asyn* vs *S-Syn*) and (2) the lighting condition (good vs. poor). We conduct two-factor ANONA with replications, followed by paired *t*-test as post-hoc analysis (significance level set at 0.01).

First, with respect to the percentage of successful face recognition, there are main effects of both factors, and no interaction effect (F (1,56) = 2.27, p = 0.134). Post-hoc analysis using paired *t*-test shows that when the lighting condition is good, the success rates of face recognition are satisfactory for both *M-Asyn* (100%, STD = 0) and *S-Syn* modes (95.7%, STD = 14.1%). The difference between the two conditions is not significant: t (29) = 1.67; p = 0.052. However, for the poor lighting condition, *M-Asyn* (mean: 89.0% STD: 17.8%) outperforms *S-Syn* (mean: 74.7%, STD: 28.4%). Paired *t*-test shows significant improvement of *M-Asyn* against *S-Syn*: t(29) = 4.69; p < 0.001.

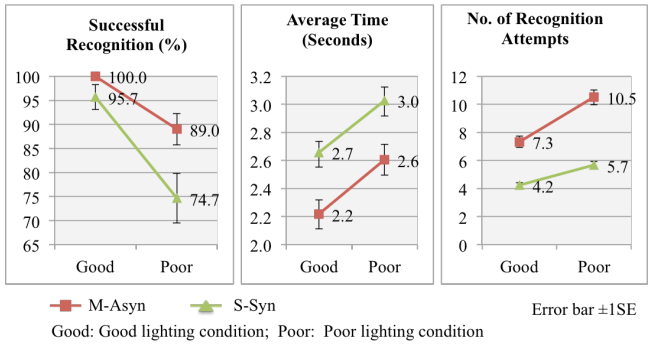

Figure 4: Performance measures of two structures.

With respect to the average time taken for a successful recognition, there are main effects of both factors, and no interaction effect (F(1,56) = 0.007, p = 0.931). The performance of the *M-Asyn* is better than *S-Syn* in both lighting conditions. In the good lighting condition, the average times of face recognition are 2.21 seconds (STD = 0.44s) for the *M-Asyn* and 2.66s (STD = 0.57s) for the *S-Syn* mode, respectively. Thus, there is significant difference between the two measures (t (29) = – 7.47, p < 0.001). When the lighting condition is poor, *M-Asyn* effectively outperforms *S-Syn*: t (29) = – 4.27, p < 0.001.

Finally, there are main effects of both factors on the number of attempts made before successful recognition, with a marginal interaction effect (F(1,56) = 5.67, p = 0.02). Under the good lighting condition,

much more recognition attempts have been made in the *M-Asyn* mode (mean = 7.33; STD = 2.16) than in the *S-Syn* model (mean = 4.23; STD = 1.05). Similar results can be found for the poor lighting condition, whereas the gap between the two conditions is even more significant. Apparently, this is a result of opportunistic multi-tasking where the server processes multiple images using multiple CPUs. This should not be interpreted as less efficient because the processing happens in parallel along different threads. Thus, the overall processing time is not increased despite the additional computational load caused by more images. In fact, this is the main factor that contributes to the higher accuracy and less recognition time of the *M-Asyn* mode. By enjoying a larger pool of input images, it increases the chance of capturing an image of better quality amid poor ones. This is particular relevant to a dynamic interaction process in inferior lighting conditions.

Figure 5: Experiment II - Interaction between two participants and two volunteers.

Experiment II – Field Test

While Experiment I shows quantitative evidence of the performance advantage of the multi-threaded asynchronous structure, a more important issue is how such performance influences user experience in a practical interaction scenario. In particular, we want to find out what is the desirable performance level in terms of speed and accuracy of face recognition, and if our system can achieve such performance needs.

To do so, we conducted a field test to evaluate the application in a real interaction scenario in the workplace. Twenty subjects participated in the study (7 female; Mean age: 24.6 yrs and standard deviation (STD): 5.2 yrs). In an experiment session, participants wore Google Glass and met with two persons (randomly chosen from the 15 volunteers in the face dataset). The Glass performed face recognition, which is used as a trigger to retrieve the personal information, i.e., upon successful recognition it retrieved the biographical information of the respective persons from the database and showed it on the Glass's prism display. Each experiment session lasted about 15 minutes. The main purpose of the meeting is for the participants to get to know the volunteers in a simulated context where they just started working in a new company (Figure 5). After the experiment session, we interviewed the subjects about their attitudes towards the device. In particular, we reviewed the instances of face recognition and retrieved the time taken for successful face recognition. In case the system fails to recognize a person or retrieves the wrong person's information, a failure is reported. Meanwhile, we collected subjects' perception of the speed of face recognition using a few questionnaire items (e.g. "I think the system is fast in recognizing the person") based on a 7-point Likert scale. (1: not at all, 7: very much so). Note that other aspects of user experience were investigated in the experiment, whereas they are not reported in this paper.

The current system achieves an accuracy of 92% (i.e., 3 failures out of 38 instances), and the speed of recognition range from 1 to 5 seconds in the given context (mean: 2.7s, STD: 0.9s). The statistics shows larger variation as compared to that in Experiment I, mainly due to the uncertainties in a real interaction context. The mean response time is 2.7 seconds (std=0.94s) and the mean acceptance level is 3.4 (std=1.66). Subjects' acceptance level is plotted in Figure 6. There is a rough trend of acceptance decline with an increase of response time. Fitting the data

points using a linear trend-line and intersecting it with an acceptance level of '4' (corresponding to *moderate level of acceptance*), an acceptable speed of face recognition is estimated to be 2.3 seconds. Apparently, there is still a sizable portion of instances where the time taken for a successful recognition is above this threshold. Nevertheless, our test happened in an unconstrained environment with varying illumination conditions and motion blur. As such, it is close to supporting practical applications in the wild, which has been barely tackled in legacy systems.

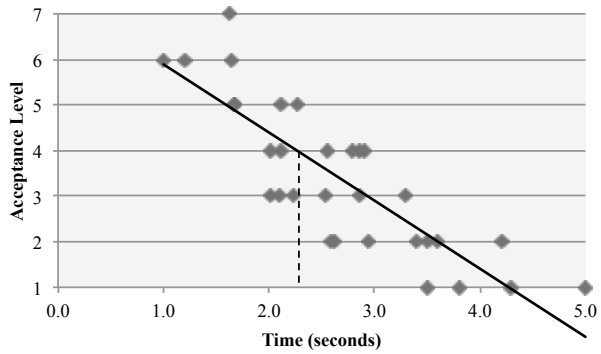

Figure 6: Data plot showing the relationship between user acceptance level and response time of face recognition. Each data point denotes an instance of face recognition. The solid line is a linear fitting of the trend. Acceptance Level '4' is considered to be a moderate level of acceptance.

A notable detrimental factor of user attitude is system failure. It is observed that once a false recognition happened, users' trust of the system plummeted. This is true for all three subjects who experienced the error. Meanwhile, the subjects were dissatisfied by the fact that only frontal face was recognizable. Since pointing the camera (on Google Glass) directly toward a person's face is impolite, it is desirable to recognize the face using the side- or oblique- view images. Some subjects suggested that face recognition be done offline, i.e. to capture a few good quality images when opportunities arise, and carry out the recognition at the back-end. This may mitigate the awkwardness of pointing or aligning the camera towards a person for a prolonged face recognition process.

Discussions

The proposed multi-threaded asynchronous structure consistently outperforms the sequential synchronous structure [5]. Therefore, it is a useful strategy to enhance the system performance in such a scenario. Nevertheless, it is worth mentioning that there is a trade of performance gain and computation overhead. First, one may expect higher computational load in the *M-Asyn* mode due to data transfer (via Bluetooth) and multi-threading. This is caused by (1) the overhead for managing and scheduling multiple threads on the server, and (2) the need to process more images in total. Thus, the structure consumes more CPU resources. Accordingly, it may lead to higher energy consumption, albeit this is not tested in the study. Second, there is an apparent increase of data transfer load (L) due to the number of threads (N). Such an increase is partially reflected by the number of recognition attempts (Figure 4). Since we did not conduct systematic study, we do not know the exact relationship between the number of threads and data transfer volume. A reasonable guess is $L \propto N^b$, where $b<1$. Second, it may cause a depletion of CPU resource if the devices need to process multiple tasks on top of face recognition. Nevertheless, for a highly dynamic

task such as social interaction, the performance factor probably outweighs energy efficiency. Third, the structure is tested on a quad-core structure, so that the results may not be generalizable to other structures with varying number of CPU cores. Therefore, it may be beneficial to consider the complexity of the applications and the server hardware before switching to multi-threaded structure.

Based on the second experiment, we observed stringent demands on the performance of face recognition in a social interaction scenario. The multi-thread asynchronous structure can partially achieve the technical demands as shown in our experimental results. However, we acknowledge that performance alone may not cover all users' concerns. An in-depth study of user needs and attitudes is required [8].

Conclusion

In this paper, we present a multi-threaded asynchronous structure that has been implemented for face recognition application in wearable devices. The experimental results show that this structure improves the application's performance by using multi-threading to process more images and return the recognition results to the client in an asynchronous manner. Although the structure is implemented on a face recognition application, we believe it can be applied to many other similar applications. Moreover, as mobile devices become more powerful, this structure has the potential to reap significant performance improvements with minimal code changes.

Acknowledgements

The work is funded by the Singapore A*STAR JCO VIP – Revers-Engineering Visual Intelligence for Cognitive Enhancement (Project No. 1335H00098).

References

1. Anam, A., Alam, S. and Yeasin, M.. Expression: A dyadic conversation aid using google glass for people with visual impairments. in *Proc. UbiComp'14*, (2014), 211-214.
2. Flautner, K., Uhlig, R., Reinhardt, S. and Mudge, T. Thread-level parallelism and interactive performance of desktop applications, in *Proc. ASPLOS-IX*, (2000),129-138.
3. Ha, K., Chen, Z., Hu, W. Richter, W., Pillai, P. and Satyanarayanan, M.. Towards wearable cognitive assistance, in *Proc. MobiSys'14*, (2014), 68-81.
4. Hwang, A.D. and Peli, E., Augmented edge enhancement for vision impairment using google glass, *SID Symposium Digest of Technical Papers*, 45, (2014), 305-307.
5. Mandal. B., Chia, S.C., Li, L., Chandrasekhar, V., Tan, C. and Lim, J.H., A wearable face recognition system on google glass for assisting social interactions, in *Proc. ACCV'14*, (2014).
6. Wang, X., Zhao, X., Prakash, V., Shi, W. and Gnawali, O., Computerized-eyewear based face recognition system for improving social lives of prosopagnosics, in *Proc. PervasiveHealth'13*, (2013),77-80.
7. Wasik, B. Why wearable tech will be as big as the smartphone, *Wired*, http://www.wired.com/2013/12/wearable-computers/ (last visited 19/06/2015)
8. Xu, Q. Mukawa, M., Li, L., Lim, J-H, Tan, T., Chia, S.C., Gan, T. and Mandal, B., Exploring users' attitudes towards social interaction assistance on google glass, in *Proc. AH'15*, (2015), 9-12.